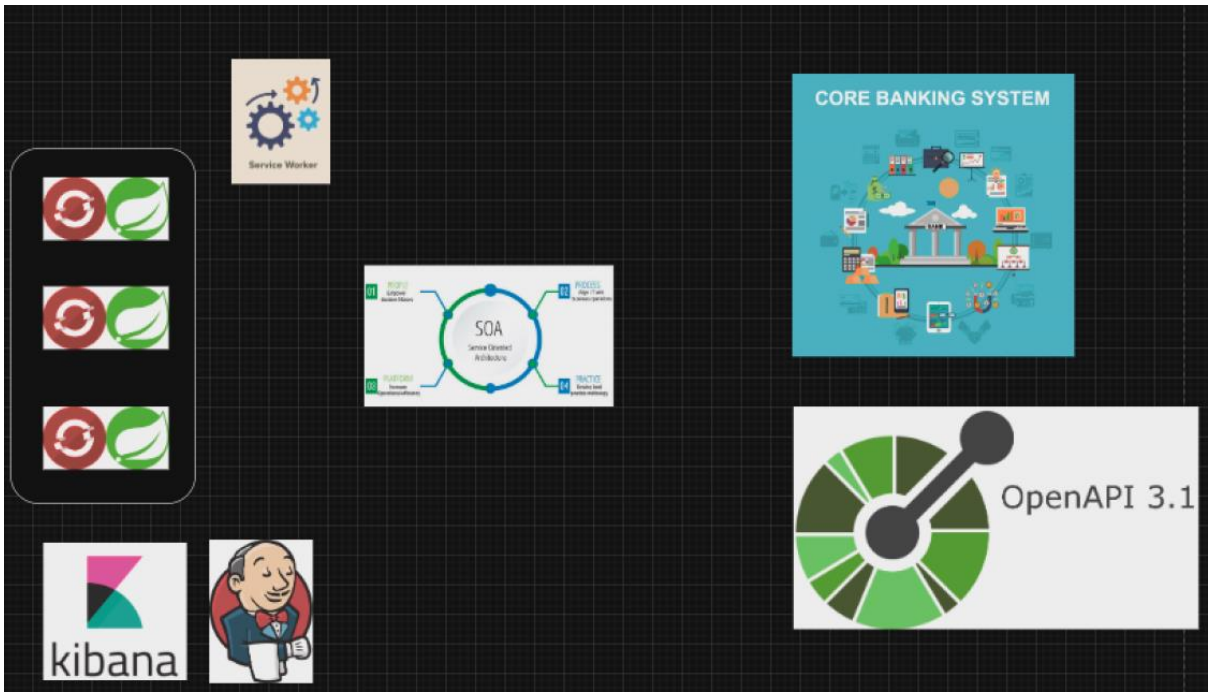
Test Integration Analyst

PT. XYZ adalah sebuah perusahaan fintech yang ingin mengembangkan mobile apps mereka, dalam Upaya menjangkau pengguna yang lebih luas mereka ingin mengembangkan aplikasi pinjaman online.

1. Coba deskripsikan tentang diagram berikut ini :



PT. XYZ telah memiliki *core banking system*. Dimana *core banking system* yang telah dimiliki, merupakan sistem *back-end* yang memproses transaksi perbankan harian dan memposting pembaruan pada rekening dan catatan keuangan lainnya. *Core banking system*, biasanya mencakup kemampuan pemrosesan simpanan, pinjaman dan kredit, dengan antarmuka ke sistem buku besar dan alat pelaporan. Dari *core banking system* tersebut akan dibuatkan *mobile apps* untuk kebutuhan pinjaman *online*.

Dengan menerapkan desain perangkat lunak arsitektur berorientasi layanan (*Service Oriented Architecture*), layanan disediakan ke komponen lain komponen aplikasi, melalui protokol komunikasi dalam jaringan. Dalam *SOA*, sejumlah layanan berkomunikasi satu sama lain, melalui salah satu dari dua cara. Yaitu melalui penyampaian data atau melalui dua atau lebih layanan yang mengoordinasikan suatu aktivitas.

Dalam berkomunikasi data antara *core banking system* dengan *mobile apps*, mempergunakan *Application Programming Interface*, yaitu mekanisme yang memungkinkan

dua komponen perangkat lunak untuk saling berkomunikasi menggunakan serangkaian definisi dan protokol. API yang dipergunakan adalah OpenApi 3.1.

Arsitektur aplikasi mempergunakan *Service worker*, yang merupakan salah satu API (*Application Programming Interface*) Javascript dimana memungkinkan pengembang dalam melakukan pemrograman *cache* dan melakukan *load asset data*, melakukan manajemen *push notification*. Dengan *service workers*, kita dapat mengatur *network requests*, *cache requests* untuk meningkatkan *performance* dan menyediakan *offline access* dengan menggunakan *cached content*.

Aplikasi yang dikembangkan berbasiskan bahasa pemrograman Java, dan mempergunakan kerangka kerja (*framework*) Spring. Spring Framework adalah kerangka aplikasi dan inversi wadah kontrol untuk *platform* Java.[2] Fitur inti kerangka kerja ini dapat digunakan oleh aplikasi Java apa pun, namun terdapat ekstensi untuk membangun aplikasi web di atas platform Java EE (Enterprise Edition). Kerangka kerja ini tidak memaksakan model pemrograman tertentu. Spring *Framework* adalah perangkat lunak gratis dan sumber terbuka.

Aplikasi mobile yang dibangun diatas dasar perangkat lunak *container*/kontainerisasi. Aplikasi yang dipergunakan untuk kontainerisasi adalah Red Hat OpenShift. Kontainerisasi ini dipergunakan, sehingga kegiatan *DevOps* antara tim operation dan tim pengembang dalam melakukan proses *build*, test dan *release software* dapat berjalan lebih efektif dan efisien.

Red Hat OpenShift merupakan *platform* aplikasi *container open source* kuat yang dirancang untuk mengotomatiskan penerapan, penskalaan, dan pengelolaan aplikasi di dalam container. Ini memberi pengembang platform lengkap untuk membangun, menerapkan, dan mengelola aplikasi di berbagai lingkungan, termasuk *cloud* publik, privat, dan hybrid.

Pada server, dipergunakan Jenkins. Yang merupakan server otomatisasi *open source*. Memungkinkan pengembang membangun, menguji, dan menerapkan perangkat lunak. Membantu mengotomatiskan bagian-bagian pengembangan perangkat lunak yang terkait dengan pembuatan, pengujian, dan penerapan, memfasilitasi integrasi berkelanjutan, dan pengiriman berkelanjutan.

Dalam melakukan indexing pada aplikasi. Dipergunakan aplikasi Kibana, untuk melakukan *Elastic Search*. Selain itu, aplikasi Kibana juga dapat dipergunakan untuk pembuatan *dashboard* dan laporan dalam bentuk grafik. Sehingga fungsi Kibana secara garis besar untuk menjelajah, memvisualisasikan dan menemukan data.

2. Buatlah rancangan spesification api yg relate untuk menghandle salah satu fitur pinjol ini (dalam bentuk apapun):

A. *Identifikasi kebutuhan fungsional*

No	Actor	Fungsi	Deskripsi
1	Nasabah	Authentication	Proses nasabah pengguna applikasi mobile untuk masuk ke dalam aplikasi.
2	Nasabah	Mengajukan pinjaman	Proses nasabah mengajukan pinjaman.
3	Nasabah	Profile nasabah	Menampilkan detil data nasabah.

B. *Analisis kebutuhan data*

- Data Nasabah

  Merupakan master data nasabah. Data nasabah yang diperlukan adalah Id, Email, Password dan Username.

- Data Peminjaman

  Merupakan table transaksi data peminjaman. Data transaksi peminjaman yang diperlukan adalah Id_transaksi, Id_nasabah, Waktu_peminjaman dan Nilai_peminjaman.

- Data pembayaran pinjaman

  Merupakan table transaksi transaksi pembayaran. Data transaksi pembayaran yang diperlukan adalah Id_transaksi, Id_transaksi_peminjaman, Nilai_pembayaran, Waktu_pembayaran.

C. *Desain API*

API Model Nasabah

No	Proses	API	
		Method	Path
1	Sign in	POST	/api/nasabah/signin
2	Fitur mengajukan pinjaman	POST	/api/nasabah/pinjam
3	Fitur melihat daftar pembayaran	GET	/api/nasabah/histori